AI-assistent för kompetensutveckling

Behovsanalys och vägledning för lärande i arbetslivet

LUNDS UNIVERSITET
Lunds Tekniska Högskola

LTH vid Campus Helsingborg

LTH Faculty of Engineering at Campus Helsingborg Department of Computer Science

Examensarbete:

Rania Alakhras

Yomna Alshehabi

LTH vid Campus Helsingborg

Lund University

Box 882

251 08 Helsingborg

Tryckt i Sverige

Lunds universitet

Lund 2026

Sammanfattning

Detta examensarbete undersöker hur en AI-assistent kan användas för att stödja behovsanalys och vägledning inom kompetensutveckling. Många organisationer och individer har svårt att snabbt identifiera vilka utbildningar som bäst motsvarar deras av kompetensbehov, särskilt när utbildningsutbudet är brett och varierat. Samtidigt uppstår ofta situationer där användarnas behov inte exakt motsvarar befintliga kursbeskrivningar. Detta skapar ett behov av intelligenta system som kan stödja användare i att formulera sina behov samt effektivt identifiera relevanta utbildningsalternativ.

Syftet med arbetet är att utforma och utveckla en fungerande prototyp av en AI-assistent som kan vägleda användare genom dialog, identifiera deras kompetensbehov och matcha dem mot relevanta kurser i kursutbudet. Om ingen direkt matchning finns ska AI-assistenten förse användaren med företagets kontaktmejl för fortsatt dialog.

Examensarbetet genomförs i samarbete med Edument AB, ett företag som erbjuder utbildningar och kompetensutveckling inom områden som mjukvaruutveckling, systemarkitektur, säkerhet, molnteknik och artificiell intelligens.

Resultatet av examensarbetet är en fungerande prototyp som demonstrerar hur AI-baserad kommunikation kan användas för att förbättra användarupplevelsen på en utbildningsplattform och samtidigt stödja företagets arbete med behovsanalys och kompetensmatchning. Studien visar att AI-assistenten kan fungera som ett effektivt gränssnitt mellan användare och organisation genom att kombinera naturlig dialog, strukturerad informationsinsamling och automatiserad vägledning.

Nyckelord: AI-assistent, kompetensutveckling, kursmatchning, regelbaserat system, RAG, användarvägledning

Abstract

This thesis investigates how an AI assistant can be used to support needs analysis and guidance in professional competence development. Many organizations and individuals face difficulties in quickly identifying which training courses best match their competence needs, especially when the range of available courses is broad and diverse. At the same time, users' needs often do not correspond exactly to existing course descriptions, creating a need for intelligent systems that can help users formulate their needs and efficiently identify relevant educational alternatives.

The purpose of this study is to design and develop a functional prototype of an AI assistant capable of guiding users through dialogue, identifying their competence needs and matching them with relevant courses in an existing course catalog. If no direct match is found, the AI assistant should provide the company's contact email to facilitate continued dialogue and further support.

The thesis is carried out in collaboration with Edument AB, a company that offers education and professional development within areas such as software development, system architecture, cybersecurity, cloud technology, and artificial intelligence.

The result of the thesis is a functional prototype that demonstrates how AI-based communication can be used to improve the user experience on an educational platform while simultaneously supporting the company's needs analysis and competence-matching processes. The study shows that the AI assistant can serve as an effective interface between users and the organization by combining natural dialogue, structured information collection, and automated guidance.

Keywords: AI assistant, competence development, course matching, rule-based system, RAG, user guidance

Förord

Vi vill rikta ett stort tack till Edument AB för möjligheten att genomföra examensarbetet samt för det stöd, engagemang och den värdefulla återkoppling som vi har fått under projektets gång. Ett särskilt tack riktas till våra kontaktpersoner och utvecklare på Edument som har bidragit med idéer, tester och synpunkter som hjälpt oss att utveckla och förbättra lösningen.

Vi vill även tacka vår handledare EsterDaniel Ytterbrink vid Lunds Tekniska Högskola för vägledning, stöd och värdefulla råd under arbetets genomförande. Tack även till examinatorn Per Andersson för återkoppling och synpunkter som bidragit till arbetets kvalitet.

Slutligen vill vi tacka familj, vänner och studiekamrater för deras stöd och uppmuntran under utbildningen samt under arbetet med detta examensarbete.

Terminologi

AI-agent: Ett system som kan fatta beslut och utföra uppgifter autonomt baserat på given input och definierade mål.

CSV-fil: En strukturerad datafil där information lagras i tabellform med kommaseparerade värden.

ChatOllama är en Python-klass i LangChain-biblioteket som fungerar som en brygga mellan koden och Ollama. Den hanterar chattformat med system- och användarmeddelanden, vilket gör det enkelt att bygga AI-appar som körs lokalt. [17]

ChatOpenAI är en klass i biblioteket LangChain som används för att ansluta till och interagera med OpenAI:s stora språkmodeller (LLM:er) via OpenAI API. [17]

Embeddings: Numeriska representationer av text, ord eller dokument som används för att mäta semantisk likhet mellan olika texter.

Flask är ett lättviktigt webbapplikationsramverk för Python, utformat för att göra det snabbt och enkelt att komma igång, med möjlighet att skala upp till mer komplexa applikationer. I detta examensarbete används Flask för att bygga det API som hanterar kommunikationen mellan frontend och backend. [8]

GPT-40 mini: Den mest kostnadseffektiva mindre modell från OpenAI.

Hallucination: När en AI-modell genererar information som verkar trovärdig men som i verkligheten är felaktig eller saknar grund i tillgänglig data.

JSON: Ett textbaserat format som används för att strukturera, lagra och utbyta data mellan olika system.

LangChain: Ett ramverk med öppen källkod för applikationsutveckling med hjälp av stora språkmodeller (LLM:er). LangChain tillhandahåller verktyg och API:er i både Python- och JavaScript-bibliotek som förenklar processen att bygga LLM-drivna applikationer som chatbotar och AI-agenter. I detta examensarbete används LangChain för att integrera RAG-systemet med kursdata och språkmodellerna. [20]

LLM (Large Language Model): En stor språkmodell som tränats på stora mängder textdata för att förstå, analysera och generera naturligt språk.

Ollama: Ett verktyg som används för att köra stora språkmodeller lokalt på en dator utan behov av molnbaserade tjänster.

Orchestrator: En samordnare som styr informationsflödet mellan olika delar i systemet och säkerställer att processerna sker i rätt ordning.

Phi3 är en liten men kraftfull AI-språkmodell utvecklad av Microsoft. Den är tränad för att vara effektiv trots sin mindre storlek, vilket gör den lämplig att köra lokalt på vanliga datorer via Ollama.

RAG (Retrieval-Augmented Generation): En teknik som kombinerar informationshämtning med textgenerering för att förbättra precisionen i AI-genererade svar genom användaning av externa datakällor.

Regelbaserat system: Ett system som styrs av fördefinierade regler, ofta i form av if-then-strukturer, för att fatta beslut eller utföra uppgifter.

Semantisk sökning: En sökmetod som baseras på innehållets betydelse snarare än enbart exakta nyckelord eller fraser.

Vektordatabas: En databas som lagrar embeddings och möjliggör snabb semantisk sökning baserat på likhet mellan vektorer.

Vite är ett byggverktyg för modern webbutveckling som består av två delar: en utvecklingsserver som möjliggör snabb uppdatering av kod under utveckling utan att hela applikationen behöver laddas om, samt ett byggsystem som paketerar och optimerar koden inför driftsättning. [7]

Innehållsförteckning

1. Inledning

Artificiell intelligens har blivit en viktig del av många företags verksamhet och används idag för att effektivisera processer, förbättra kundservice samt stödja beslutsfattande [13]. En vanlig tillämpning är AI-assistenter, som kan kommunicera med användare genom naturligt språk och snabbt ge information, vägledning och rekommendationer.

För företag kan AI-assistenter bidra genom att minska manuellt arbete, spara tid och ge stöd till kunder dygnet runt utan att personal alltid behöver vara tillgänglig. Forskning visar även att AI-assistenter kan förbättra kundnöjdhet och öka konverteringsfrekvenser inom digitala tjänster [19]. Inom utbildning och kompetensutveckling kan en AI-assistent hjälpa användare att identifiera sina behov, hitta relevanta kurser samt skapa en smidigare kontakt mellan användare och företag.

Ett exempel på ett företag som ser potential i AI-assistenten inom kompetensutveckling är Edument AB, som erbjuder utbildningar inom IT och teknik. Detta examensarbete genomfördes i samarbete med Edument AB och fokuserar på att utveckla en AI-assistent som stödjer behovsanalys och vägleder för användare som söker relevanta kurser. Målet är att skapa en lösning som kan förbättra både användarupplevelsen och företagets arbete med kompetensmatchning.

För att undersöka vilken lösning som bäst uppfyllde projektets krav utvärderades och jämfördes flera tekniska tillvägagångssätt under arbetets gång, bland annat RAG-teknik och regelbaserade system. Genom denna utvecklingsprocess kunde olika lösningars styrkor och svagheter analyseras med fokus på svarstid, flexibilitet, behovsanalys och kursmatchning.

1.1 Bakgrund

Detta examensarbete genomförs på uppdrag av Edument AB, ett företag som erbjuder utbildningar inom områden såsom mjukvaruutveckling, cybersäkerhet, molnteknik och artificiell intelligens. Företagets breda kursutbud skapar utmaningar. När antalet tillgängliga kurser är stort kan det vara svårt för användare att snabbt och självständigt identifiera vilken utbildning som bäst svarar mot deras specifika behov. Idag kräver detta ofta manuell kontakt med företaget, vilket är tidskrävande för både användaren och Eduments personal.

Ytterligare en utmaning är att användarnas kompetensbehov ofta är komplexa och inte alltid kan kopplas direkt till en specifik kurs. Behoven kan omfatta flera kompetensområden samtidigt, samtidigt som användarna inte alltid känner till den terminologi som används i kurskatalogen. Detta försvårar identifieringen av relevanta utbildningar genom traditionella sök- och filtreringsfunktioner.

Examensarbetet syftar till att lösa detta genom att utveckla en AI-assistent som kan analysera användarens behov, rekommendera relevanta kurser samt skapa fortsatt kontakt om ingen passande kurs finns. AI-assistenten är utformad för att inte bara svara på direkta kursfrågor utan även föra en dialog med användaren för att identifiera och förstå deras kompetensbehov. Om ingen lämplig kurs hittas samlas konversationen och användarens kontaktuppgifter in och vidarebefordras till Edument för vidare hantering.

1.2 Syfte

Syftet med examensarbetet är att utveckla en AI-baserad assistent som kan:

- guida webbplatsbesökare till relevanta utbildningar
- hjälpa användare att tydliggöra sitt behov genom dialog
- samla in strukturerade förfrågningar när färdig kurs inte finns
- fungera som ett första steg i Eduments behovsanalys- och säljprocess.

1.3 Mål

Examensarbetet syftar till att designa, implementera och utvärdera en fungerande prototyp av en AI-assistent för både backend och frontend. AI-assistenten ska analysera användarens kompetensbehov och matcha dessa mot det mest relevanta alternativet i kursutbudet. Om ingen lämplig matchning identifieras vidarebefordras behovet till Edument för vidare hantering.
Examensarbetet ska uppnå följande mål:

- Utveckla en fungerande prototyp
- Korrekt behovsanalys : minst 80 % av testfallen ska resultera i korrekt identifierade behov.
- Matchning mot kursutbud : AI-assistenten ska kunna matcha användarens behov mot
  Eduments kursutbud i minst 80 % av fallen.
- Hantering av icke-matchande behov :
    - Informera användaren tydligt.
    - Meddela Edument att kursen saknas.

- Användargränssnitt: Minst 5 testanvändare ska kunna använda systemet utan handledning och genomföra en fullständig interaktion.
- Prestanda: AI-assistenten ska ge svar inom max 15 sekunder vid normal användning (lokal körning).
- Utvärdering: Prototypen ska testas genom:
    - Intern testning (utvecklare)
    - Minst 5 externa användare

1.4 Problemformulering

Hur kan en AI-assistent designas och implementeras för att:

- förstå användarens kompetensbehov?
- matcha dessa behov mot ett befintligt kursutbud?
- identifiera när behovet inte matchar befintliga kurser?
- skapa ett effektivt gränssnitt mellan användare och företag när en matchning saknas?

1.5 Motivering av examensarbetet

Intresset för artificiell intelligens och naturlig språkbehandling (NLP) var avgörande vid valet av examensarbete. Tidigare kunskaper inom maskininlärning i kursen projekt årskurs 3 på universitet, gjorde området särskilt motiverande att fördjupa sig inom.

För Edument kan AI-assistenten hjälpa besökare dygnet runt att snabbt hitta rätt utbildning utan att behöva kontakta företaget via telefon eller e-post, vilket sparar både tid och kostnader. Dessutom kan AI-assistenten registrera frågor och behov som Edument för närvarande inte erbjuder kurser inom, vilket kan hjälpa företaget att utveckla nya utbildningar i framtiden.

När företag som Edument ser konkreta fördelar med AI-assistenter, såsom effektivare kundservice, kan det motivera andra företag att göra samma sak. Forskning visar att AI-chatbotar kan förbättra kundnöjdhet och öka konverteringsfrekvenser inom digitala tjänster [19]. Detta bidrar till för samhällsnytta i stort.

1.6 Avgränsningar

Fokus ligger på prototypnivå och inte på fullskalig produktion.

AI-assistenten är avgränsad till frågor som rör Eduments kursutbud och testas endast på svenska.

Säkerhet och dataskydd behandlas på konceptuell nivå.

1.7 Arbetsfördelning

	Rania Alakhras	Yomna Alshehabi
Analys	60	40
Design	50	50
Implementation/Konstruktion	50	50
Testning/Utvärdering	40	60
Rapport och presentation	50	50
Poster	50	50

2. Teknisk bakgrund

2.1 AI-pyramiden och AI-agentens komponenter

För att förstå hur en AI-assistent fungerar kan man använda en modell som kallas AI-pyramiden. Modellen visar hur en AI-assistent är uppbyggd, från de enklaste funktionerna i botten till de mer avancerade funktionerna längst upp.

Wang et al. [14] beskriver att AI-agenter består ofta av flera samverkande komponenter såsom språkmodell, minne, kontext och handlingar.

LLM (Large Language Model) är den centrala delen och finns i botten av pyramiden. Det är LLM:en som gör att AI-assistenten kan förstå vad besökaren skriver och svara på ett naturligt sätt. I detta examensarbete används en LLM för att förstå vilken kurs användaren behöver.

Context (kontext) är all information som AI-assistenten får tillgång till när den ska svara på en fråga. Det är inte bara användarens fråga, utan ett helt "paket" av information som skickas till språkmodellen. I detta examensarbete består kontexten av tre delar: en systemprompt med instruktioner och regler för hur AI-assistenten ska bete sig, Eduments kursutbud som hämtas dynamiskt via RAG, samt konversationshistoriken som gör att AI-assistenten kan följa samtalets röda tråd. Tillsammans gör dessa tre delar att AI-assistenten kan svara korrekt och relevant, utan att hitta på information som inte finns i kursutbudet.

Memory (minne) gör att AI-assistenten kan komma ihåg vad som sagts tidigare i samma konversation. Det betyder att användaren inte behöver upprepa sig, och att AI-assistenten kan följa konversationen.

Actions (handlingar) finns längst upp i pyramiden och är det som AI-assistenten utför baserat på vad användaren efterfrågar. I detta examensarbete kan AI-assistenten söka efter relevanta kurser, rekommendera användaren och om användaren vill kontakta Edument om ingen kurs matchar, skickar AI-assistenten konversationen och användarens kontaktuppgifter till Edument.

Tillsammans gör dessa fyra delar att AI-assistenten kan föra en naturlig dialog med användaren, förstå vad användaren behöver och ge relevanta rekommendationer.

2.2 RAG-system (Retrieval-Augmented Generation)

Enligt Microsoft Azures webbplats är retrieval-augmented generation (RAG) en AI-teknik som kombinerar en hämtningsmodell med en generativ modell [4]. Den hämtar relaterad information från en databas eller dokumentuppsättning och använder den för att generera mer exakta och kontextuellt relevanta svar. Metoden förbättrar kvaliteten på AI-genererad text genom att grunda den i verkliga data, vilket gör den särskilt användbar för uppgifter som att besvara frågor, sammanfatta och generera innehåll.

Fördelar med RAG-tekniken

- **Ökad noggrannhet:** Genom att integrera extern kunskap minskas risken för så kallade hallucinationer, där modellen skapar rimliga men felaktiga påståenden.
- **Transparens och förtroende:** Genom att kunna ange källor för sin information skapar RAG-tekniken ett system där användaren kan verifiera svaren, vilket är grundläggande för mänskligt förtroende.
- **Sammanhangsberoende:** Systemet blir bättre på att förstå den specifika situationen eller branschen, vare sig det gäller juridik, medicin eller utbildning.

Processen i RAG-system

Processen börjar med att extern data samlas in och omvandlas till numeriska representationer, så kallade embeddings, som lagras i en vektordatabas. När en användare ställer en fråga omvandlas även frågan till en liknande representation, vilket gör att systemet kan hitta de mest relevanta dokumenten genom semantisk sökning.

Den insamlade informationen kombineras därefter med användarens fråga och skickas till språkmodellen, som använder detta underlag för att generera ett mer korrekt, relevant och faktabaserat svar. Enligt AWS [5] kan modellen arbeta med aktuell

information genom kontinuerliga uppdateringar av databasen, vilket gör RAG användbart inom områden där hög precision krävs.

En viktig fördel med RAG är att ny information kan göras tillgänglig för systemet utan att språkmodellen behöver tränas om. Om kursutbudet förändras kan nya eller uppdaterade kursbeskrivningar läggas till i datakällan och därefter indexeras i vektordatabasen. AI-assistenten kan då använda den uppdaterade informationen vid nästa sökning. Detta gör RAG särskilt lämpligt för tillämpningar där informationen förändras över tid, exempelvis utbildningskataloger där nya kurser tillkommer och befintliga kurser uppdateras.

2.3 Pandas-bibliotek

Pandas är ett Python-bibliotek för hantering och analys av strukturerad data, såsom tabeller och CSV-filer. Istället för att omvandla text till vektorer arbetar Pandas direkt mot datafiler och kan filtrera och söka i dem baserat på nyckelord och kategorier. Detta gör Pandas till ett enkelt och snabbt alternativ när datan redan är strukturerad och avancerad semantisk sökning inte krävs.

2.4 Regelbaserat system

Ett annat vanligt tillvägagångssätt vid utveckling av chatbots är användning av regelbaserade system. Enligt SAP [6] kommunicerar regelbaserade AI-assistenter med hjälp av fördefinierade regler programmerade av utvecklaren.

Systemet styrs av fördefinierade regler och villkor där chatbotens beteende bestäms i förväg av utvecklaren. Systemet följer en enkel logik där specifika användarfrågor kopplas till bestämda svar, enligt principen:

> “Om användaren säger X → svara Y”

Ett praktiskt exempel är en sjukvårdsbot, där besökare skriver “Vad gör jag vid feber?” och systemet känner igen nyckelord som “feber” och svarar med förutbestämda råd, oavsett hur frågan är formulerad.

Teknisk uppbyggnad:

Regelbaserade system bygger tekniskt på två huvudkomponenter. Den första är mönstermatchning, där systemet känner igen nyckelord i användarinmatningar och matchar dem med ett specifikt svar. Den andra är beslutsträd, där systemet följer en trädstruktur av if-then-regler för att hitta rätt svar. Till exempel:

- Om användaren skriver "feber" → fråga om temperaturen
- Om temperaturen är över 39°C → rekommendera läkarkontakt
- Om temperaturen är under 39°C → ge råd om egenvård

Fördelar med regelbaserade system:

- Hög kontroll och förutsägbarhet: Utvecklaren bestämmer exakt vad chatboten svarar, vilket minskar risken för felaktiga eller olämpliga svar.
- Enkelt att implementera: Systemet kräver ingen avancerad AI.
- Lämpar sig för begränsad data: När informationsmängden är liten och inte förändras ofta kan regelbaserade system hantera alla frågor effektivt utan behov av avancerad teknik.

Hantering av ny information:

En begränsning med regelbaserade system är att ny information normalt måste lägga till manuellt genom att regler, nyckelord eller svar uppdateras av utvecklaren. Om kursutbudet förändras krävs därför vanligtvis manuella ändringar i systemet för att den nya informationen ska kunna användas.

3. Metod

I detta examensarbete följdes en iterativ utvecklingsprocess där tekniska beslut fattades successivt baserat testresultat och återkoppling. Processen delades in i tre faser: inlärningsfas, implementationsfas och testningsfas. Under implementationsfasen identifierades konkreta begränsningar med den initialt valda tekniken, RAG med phi3 via Ollama, vilket ledde till ett motiverat teknikbyte. Det andra tekniska valet, regelbaserad teknik, utvärderades därefter under testningsfasen tillsammans med Eduments utvecklare, varvid ytterligare begränsningar identifierades som ledde till ett nytt beslut. Det slutliga systemet baseras på RAG-teknik med GPT-4o mini, vilket rekommenderades av Eduments utvecklare baserat på de observerade bristerna. Denna iterativa process är i linje med hur praktisk systemutveckling ofta bedrivs, där tekniska val kontinuerligt utvärderas mot uppsatta krav.

3.1 Inlärningsfas

3.1.1 Backend

Arbetet påbörjades med en förstudie för att skapa en förståelse av vad en AI-assistent är och vilka teknologier som skulle användas. För att förstå hur en lokal AI-agent kan byggas användes videoguider och praktiska demonstrationer [1]. Därefter genomfördes en undersökning av olika AI-assistenter på flera myndigheters webbplatser, däribland Skatteverket och Helsingborgs Stad. Anledningen var att undersöka olika AI-assistenter i olika sammanhang och med olika arbetsuppgifter.

En CSV-fil upprättades med beskrivningar och detaljer från Eduments kursutbud [3]. Filen lästes in via LangChain med stöd av Pandas [2] och användes som kontext för att ge AI-assistenten tillgång till kursutbudet.

3.1.2 Frontend

I denna fas undersöktes olika metoder för att integrera en AI-assistent på en webbplats. Inspiration hämtades från HubSpot, som rekommendation från utvecklaren i företaget, där AI-assistenten laddades via ett externt skript och visades i en iframe. Även grunderna i React, Vite och JavaScript samt kommunikationen mellan frontend och backend via API-anrop studerades för att skapa en grund inför implementationsfasen.

3.2 Implementation- och testningsfas

3.2.1 Backend

Arbetet fortsatte med implementering av en AI-assistent baserad på kunskap och de erfarenheter från inlärningsfasen. Målet var att utveckla en fungerande prototyp som kunde analysera användarfrågor och matcha användarens kompetensbehov mot relevanta kurser och kursinformation.

Implementationsfasen inleddes med utveckling av en RAG-baserad lösning där en vektordatabas kopplades samman med språkmodellen för att möjliggöra att hämta av relevant information vid generering av svar. I denna lösning användes orchestratorn som en samordnande komponent mellan användaren, språkmodellen och vektordatabasen för att säkerställa att rätt information skickades vidare vid rätt tidpunkt.

Under utvecklingsarbetet testades flera lokala språkmodeller via Ollama, däribland Phi-3 Mini (3.8B parametrar, cirka 2,2 GB) och

jobautomation/OpenEuroLLM-Swedish:latest (8.1 GB). OpenEuroLLM-Swedish inkluderades i utvärderingen eftersom modellen är utvecklad för europeiska språk och därför bedömdes vara relevant att jämföra med Phi3 i projektets svenskspråkiga sammanhang. Modellerna användes för att undersöka hur olika språkmodeller påverkade AI-assistentens språkförståelse, svarskvalitet och svarstid vid lokal körning.

Systemet testades på tre olika datorer för att undersöka hur hårdvarans prestanda påverkade svarstiden.

Tabell 1: Testdatorernas hårdvaruspecifikationer

Dator	Processor	RAM	Grafik
Dator 1	AMD Ryzen 7 4700U (2,00 GHz)	16 GB	Radeon Graphics (integrerad)
Dator 2	Intel Core i5-7200U (2,50 GHz)	8 GB	Intel HD Graphics 620
Dator 3			

Systemet testades på tre olika datorer för att undersöka hur hårdvarans prestanda påverkade svarstiden (tabell 1). Ingen av testdatorerna hade en dedikerad GPU, vilket innebar att språkmodell och informationshämtningen bearbetades lokalt med hjälp av datorernas processorer och arbetsminne.

För att utvärdera lösningen genomfördes tester där olika användarfrågor skickades till systemet och svarstiden registrerades. Testerna omfattade både specifika kurserfrågor och öppna frågor om kompetensbehov. Resultaten från testningen användes som underlag för vidare utveckling och jämförelse mellan olika tekniska lösningar.

Efter implementeringen av den första RAG-baserade lösningen utvecklades även ett regelbaserat system för att möjliggöra jämförelse mellan olika tekniska tillvägagångssätt. Det regelbaserade systemet implementerades för att hantera kursrelaterade frågor genom fördefinierad logik och nyckelordsbaserad matchning. Systemet följde en fördefinierad prioriteringsordning där varje inkommande fråga testades mot flera steg i tur och ordning. Först kontrollerades om frågan var en enkel dialog såsom hälsningar eller tackfraser. Därefter identifierades faktafrågor om exempelvis pris, datum eller längd baserat på nyckelordsmatchning. I ett tredje steg undersöktes om frågan innehöll ett känt teknikområde som React, Python eller Java. Om frågan innehöll ord som 'nybörjare' hanterades den i ett fjärde steg med rekommendationer av introduktionskurser. I ett femte steg genomfördes en bredare

poängbaserad sökning mot hela kursutbudet. Endast om inget av dessa steg gav ett resultat skickades frågan vidare till Phi3 via Ollama för att generera ett svar.

Det regelbaserade systemet testades genom samma typer av användarfrågor som användes vid testning av RAG-systemet. Testerna omfattade frågor om specifika kurser, öppna frågor om kompetensbehov samt situationer där ingen passande kurs fanns i kursutbudet. Även svarstiden registrerades och jämfördes mellan de olika lösningarna.

Frontend- och backendtestning genomfördes därefter tillsammans med utvecklare på Edument AB. Testerna inkluderade även frågor och användarscenarier som inte hade identifierats under den interna testningen. Baserat på resultaten från dessa tester vidareutvecklades systemet med en RAG-baserad lösning där GPT-4o mini användes som språkmodell i den slutliga implementationen.

I den slutliga lösningen implementerades även en funktion för hantering av situationer där ingen lämplig kurs kunde identifieras. I dessa fall presenterades ett formulär där användaren kunde ange namn och e-postadress. Formuläret användes för att vidarebefordra användarens kontaktuppgifter tillsammans med konversationen till Edument för fortsatt hantering.

3.2.2 Frontend

För frontend användes React tillsammans med Vite för att bygga AI-assistentens användargränssnitt som en fristående webbaseras applikation. Syftet var att skapa en enkel och tydlig chatt där användaren kunde ställa frågor och få svar direkt på webbsidan. Kommunikationen mellan frontend och backend skedde via HTTP-anrop till ett Flask API, där användarens frågor skickades vidare till AI-assistenten och svaren visades i gränssnittet. För att koppla AI-assistenten till testsidan användes även JavaScript för att skapa ett loader-skript som öppnade AI-assistenten i en iframe, vilket valdes eftersom det gjorde lösningen säkrare, flexibel och enklare att underhålla utan att påverka webbplatsens övriga struktur eller design.

Frontend testades genom användartester där både utvecklarna och externa testanvändare fick använda AI-assistenten via webbgränssnittet. Testerna fokuserade på om användaren kunde genomföra en fullständig interaktion utan handledning samt om gränssnittet upplevdes som tydligt och enkelt att använda.

Testningen genomfördes även tillsammans med utvecklare på Edument AB, där ytterligare frågetyper och användarscenarier testades. Dessa tester användes som underlag för vidare utveckling av systemet. I den slutliga implementationen användes

en RAG-baserad lösning tillsammans med GPT-4o mini som språkmodell för att förbättra hanteringen av varierande och öppna frågor från användare.

3.3 Kommunikation

Kommunikation med Edument AB:
Kommunikationen med företaget Edument skedde kontinuerligt med både teknisk handledare och icke-tekniska handledare, genom ett distansmöte varje vecka. Samtidigt fanns en grupp på WhatsApp för uppdateringar och akuta frågor.

Kommunikation med handledare vid universitetet:
Kommunikationen med handledaren skedde kontinuerligt, ett distansmöte varje vecka. Samtidigt fanns en Discord-server för uppdateringar och akuta frågor.

3.4 Källkritik

Källorna som använts i examensarbetet har valts eftersom de ger relevant information om de tekniker och metoder som används i projektet. Källorna består av dokumentation, företagswebbplatser, vetenskapliga artiklar och tekniska webbresurser.

Dokumentationen från **Pandas**, **Flask** och **Vite** bedöms som tillförlitliga eftersom de kommer från respektive verktygs officiella dokumentation. Dessa källor används för att beskriva hur teknikerna fungerar och hur de kan användas i implementationen. Eftersom dokumentationen uppdateras av utvecklarna bakom verktygen anses informationen vara aktuell och tekniskt korrekt.

Informationen från **Eduments webbplats** används för att beskriva företagets kursutbud. Denna källa är relevant eftersom examensarbetet genomförs i samarbete med Edument och AI-assistenten bygger på företagets egna utbildningar. Källan kan därför anses trovärdig för information om kurserna, men den är samtidigt en företagskälla och kan vara utformad i marknadsföringssyfte.

Källorna från **Microsoft Azure** och **AWS** används för att förklara RAG-teknik. Dessa företag är stora och etablerade aktörer inom molntjänster och artificiell intelligens. Informationen bedöms därför som relevant och tillförlitlig för en teknisk översikt. Samtidigt är dessa källor inte vetenskapliga artiklar, vilket innebär att de främst används för grundläggande teknisk bakgrund och inte som ensam grund för analys.

Källan från **SAP** används för att beskriva chatbots och regelbaserade system. SAP är ett etablerat företag inom affärssystem och digitala lösningar, vilket gör källan relevant. Eftersom informationen kommer från ett företag bör den dock tolkas som en översiktlig källa snarare än en oberoende vetenskaplig källa.

De vetenskapliga artiklarna, exempelvis **Lewis et al. (2020)**, **Gao et al. (2023)**, **Følstad och Brandtzæg (2017)**, **Gnewuch et al. (2018)** och **Wang et al. (2024)**, bedöms som särskilt tillförlitliga eftersom de är akademiska publikationer. Dessa källor används för att stärka den teoretiska bakgrunden och diskussionen kring RAG, AI-agenter, chatbots och svarstid i människa–AI-interaktion. Vetenskapliga artiklar har alltid granskats av andra forskare innan publicering, vilket ökar deras trovärdighet.

Källan från **EU-kommissionens riktlinjer för tillförlitlig AI** används i avsnittet om etiska aspekter. Den bedöms som trovärdig eftersom den kommer från en offentlig europeisk institution och behandlar etiska principer för AI, såsom transparens, mänsklig kontroll och ansvar.

YouTube-källan användes endast som stöd i inlärningsfasen för att få en praktisk introduktion till hur en lokal AI-agent kan byggas. Denna typ av källa bedöms som mindre akademiskt tillförlitlig än vetenskapliga artiklar och officiell dokumentation, eftersom innehållet inte alltid är granskat. Därför används den främst som inspiration och praktiskt stöd, inte som huvudkälla för slutsatser.

3.5 Metodutvärdering

Metoden som användes i examensarbetet var lämplig eftersom arbetet genomfördes i flera tydliga faser: inlärningsfas, implementationsfas och testningsfas. Detta gjorde det möjligt att först skapa förståelse för AI-assistenter, RAG-teknik, regelbaserade system, backend, frontend och integration mellan olika systemdelar. Därefter kunde kunskapen användas praktiskt för att utveckla och testa en fungerande prototyp.

En styrka med metoden var att tre olika tekniska lösningar utvecklades och jämfördes: RAG med phi3 via Ollama, ett regelbaserat system samt RAG med GPT-4o mini. Genom att testa och utvärdera samtliga lösningar kunde arbetet inte bara visa vilken lösning som fungerade bäst, utan också motivera varför den ena lösningen var mer lämplig än de andra. En annan styrka var att testningen omfattade både funktionalitet och användarupplevelse. Genom manuella testfall kunde systemets förmåga att identifiera behov, matcha kurser och hantera fall där ingen kurs fanns kontrolleras. Användartesterna bidrog också till att visa om gränssnittet var enkelt att använda utan handledning.

En begränsning med metoden var att antalet testanvändare var relativt litet. Resultaten ger därför en god bild av prototypens funktion, men kan inte fullt ut generaliseras till alla typer av användare. En annan begränsning var att systemet testades på prototypnivå och i en lokal miljö, vilket innebär att ytterligare tester krävs innan systemet kan användas i fullskalig produktion.

4. Resultat och analys

Under projektets gång utvärderades tre olika tekniska lösningar: RAG med phi3 via Ollama, ett regelbaserat system samt RAG med GPT-4o mini. Varje lösning testades och utvärderades innan nästa utvecklades, vilket skapade en iterativ process där resultaten från en lösning låg till grund för efterföljande tekniska beslut. I detta kapitel presenteras resultaten från dessa tre lösningar, följt av en jämförande analys av svarskvalitet och svarstid samt en utvärdering av hur väl examensarbetets krav uppfylls.

4.1 RAG-teknik med phi3 via Ollama

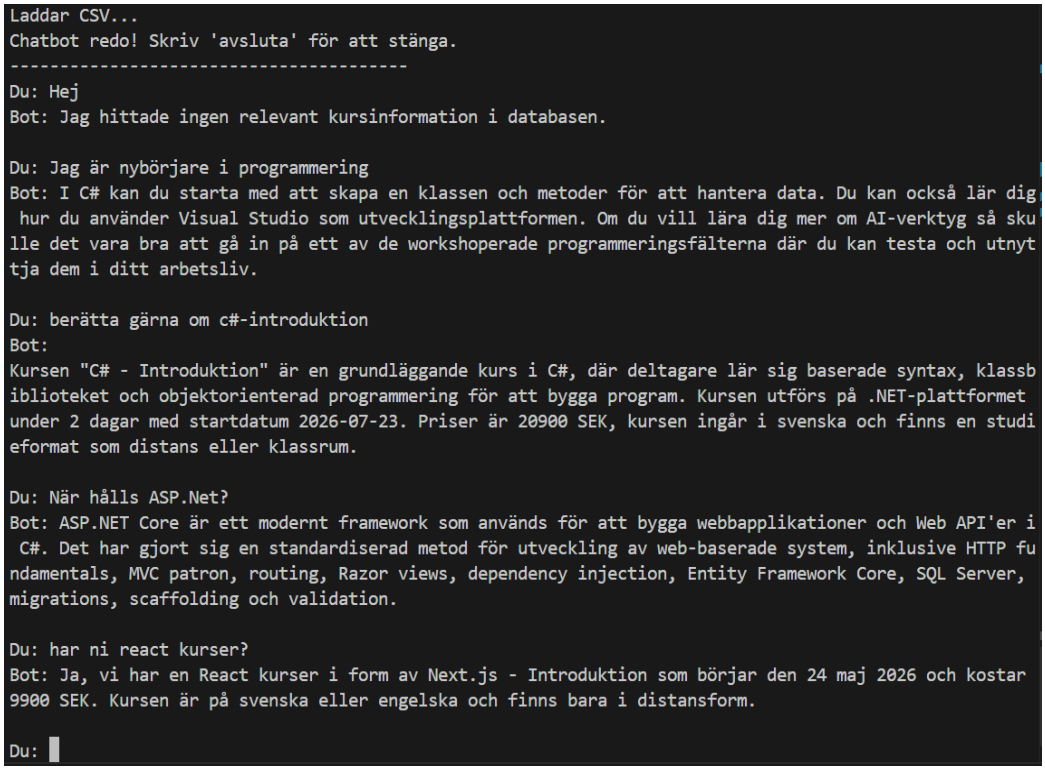
Den första implementationen baseras på RAG-teknik tillsammans med lokala språkmodeller via Ollama. Initialt används språkmodellen Phi-3 Mini (3.8B parametrar, cirka 2,2 GB) för att undersöka hur en lokal språkmodell kan hantera kursrelaterade frågor och behovsanalys inom AI-assistenten.

Under testningen observeras att svaren från phi3 inte alltid är språkligt naturliga på svenska, vilket även framgår i figur 1. För att undersöka om språkförståelsen kunde förbättras testas även den svenska språkmodellen jobautomation/OpenEuroLLM-Swedish:latest (8.1 GB).

Utöver Phi3 testades även OpenEuroLLM-Swedish via Ollama. Modellen är större än Phi3 (8,1 GB jämfört med 2,2 GB) och valdes för att undersöka om en svenskspråkigt anpassad modell kunde vara lämplig för projektet. Testerna visade att modellen kunde användas i samma RAG-arkitektur som Phi3, men den högre modellstorleken medförde större krav på lokal hårdvara. Eftersom fokus senare flyttades till jämförelsen mellan den lokala lösningen och GPT-4o mini genomfördes ingen djupare utvärdering av OpenEuroLLM-Swedish.

Systemet testas med olika typer av användarfrågor, både kursspecifika frågor och öppna frågor om kompetensbehov. Testerna genomförs på tre olika datorer utan dedikerad GPU. Resultaten visar att systemet kan generera relevanta svar, men att svarstiden varierar mellan cirka en minut och två och en halv minuter vid lokal körning.

I vissa testfall identifierar systemet rätt kursområde men ger inte ett direkt svar på användarens fråga. Exempelvis besvaras frågan ”När hålls ASP.Net?” med generell kursinformation istället för specifik information om datum eller kurstillfälle. Detta visar att systemet kan hämta relevant kursrelaterad information men inte alltid tolka användarens intention korrekt.

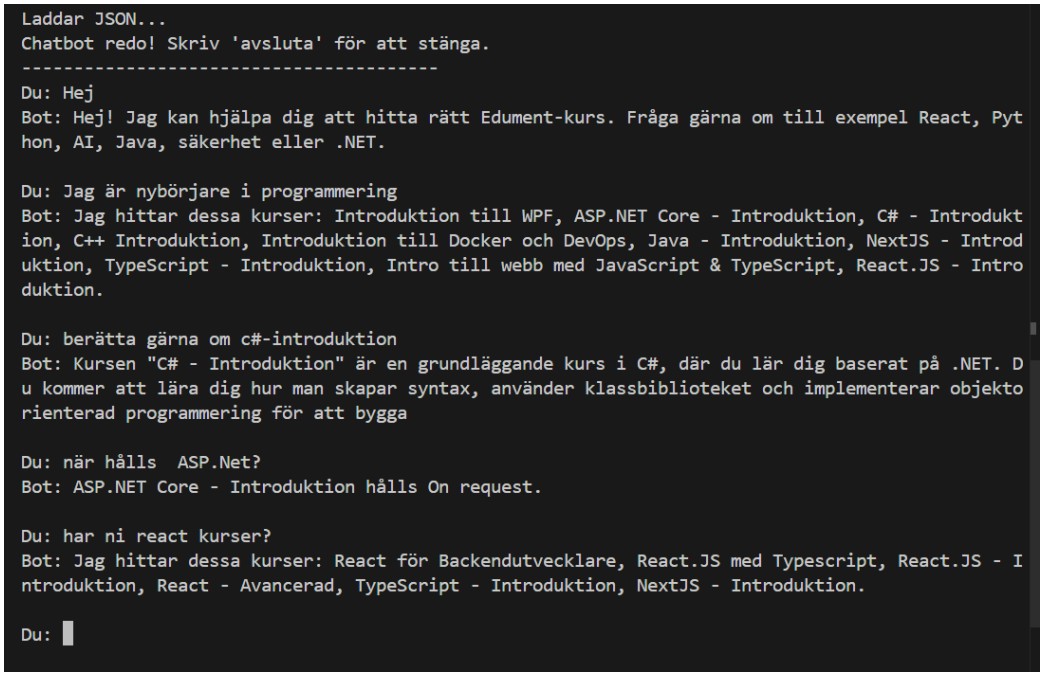


Figur 1. Visar ett exempel på konversation mellan användare och AI-assistenten i den RAG-baserade implementationen med phi3 via Ollama.

4.2 Regelbaserad teknik

Efter den första RAG-baserade implementationen, utvecklas ett regelbaserat system för att jämföra olika tekniska tillvägagångssätt. Systemet bygger på fördefinierad logik och nyckelordsbaserad matchning för att hantera kursrelaterade frågor.

Resultaten visar att systemet ger kortare och mer direkta svar vid frågor om specifika kurser. Svarstiden för kursrelaterade frågor ligger i millisekunder, medan öppna frågor som kräver användning av phi3 via Ollama ger längre svarstider.



Figur 2. Utdrag ur konversation mellan användare och AI-assistenten i den regelbaserade implementationen.

4.3 RAG-teknik med GPT-4o mini

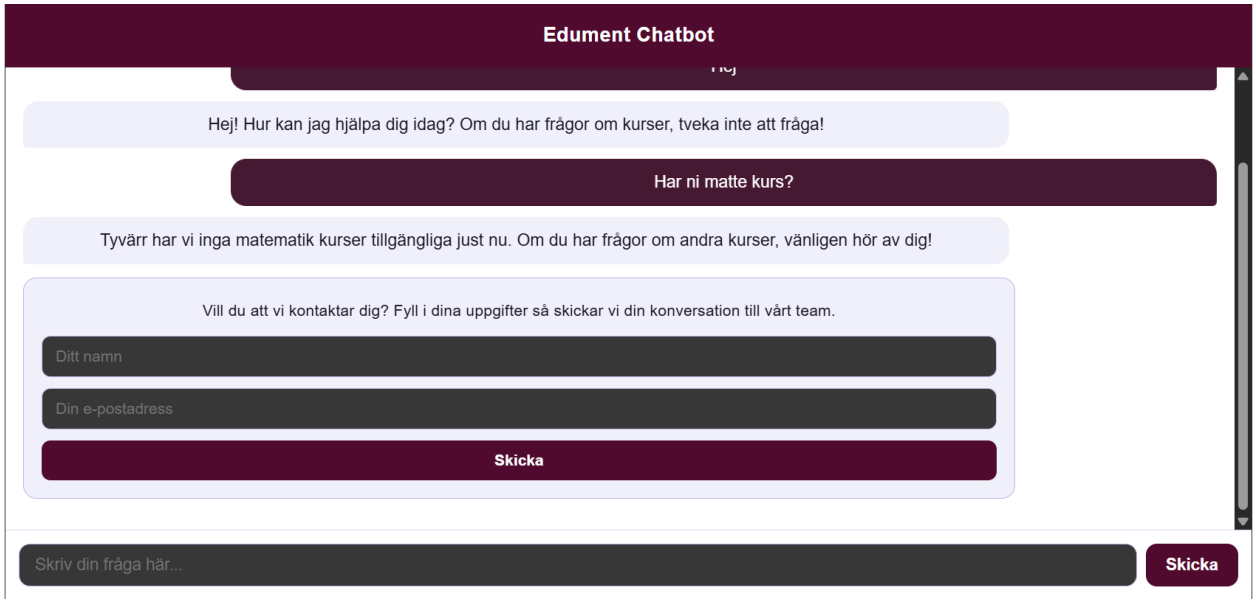
Det slutliga systemet implementeras med RAG-teknik med GPT-4o mini som språkmodell. Det är viktigt att notera att RAG-arkitekturen behölls även i den slutliga lösningen. Teknikbytet bestod främst i att de lokalt körda språkmodellerna via Ollama (Phi3 och OpenEuroLLM-Swedish) ersattes med GPT-4o mini via OpenAI API.

Användarens fråga användes fortfarande för att hämta relevanta kursbeskrivningar från kunskapsbasen genom semantisk sökning. Den hämtade informationen skickades därefter tillsammans med användarens fråga till GPT-4o mini, som genererade det slutliga svaret. På så sätt baserades svaren på det kursunderlag som lagrats i systemets kunskapsbas snarare än enbart på språkmodellens förtränade kunskap.

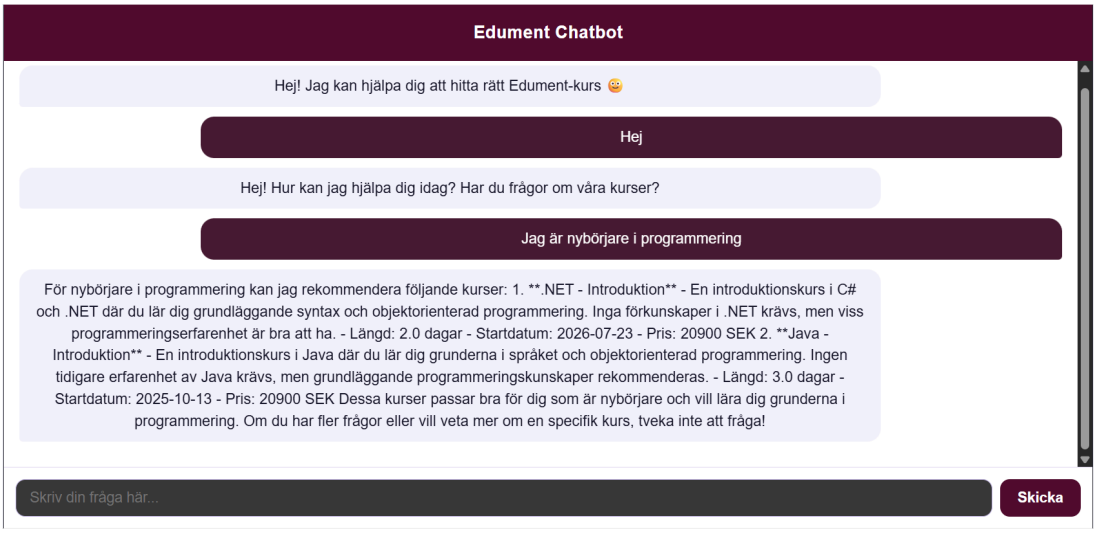
Systemet hanterar både kursinriktade frågor och oväntade frågetyper på ett tillfredsställande sätt. Svarstiden uppfyller kravet på maximalt 15 sekunder och svaren är språkligt naturliga och relevanta.

Systemet hanterar även situationer där ingen lämplig kurs kan identifieras. I dessa fall erbjuds användaren möjlighet att fylla i sitt namn och sin e-postadress för fortsatt dialog med Edument. När formuläret skickas vidarebefordras både användarens

kontaktuppgifter och den aktuella konversationen till Edument. På så sätt kan företagets personal ta del av användarens behov och fortsätta dialogen även när en direkt kursmatchning saknas. Funktionen illustreras i figur 3.

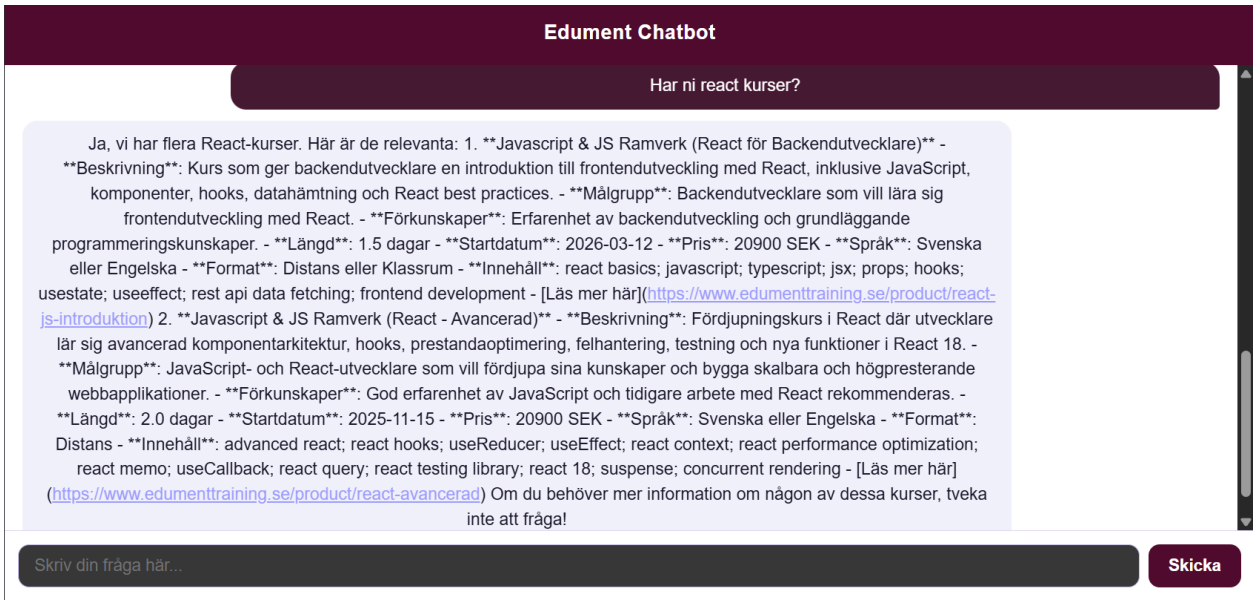


Figur 3. Bilden visar konversationen mellan användare och AI-assistenten i RAG-teknik med GPT-4o mini

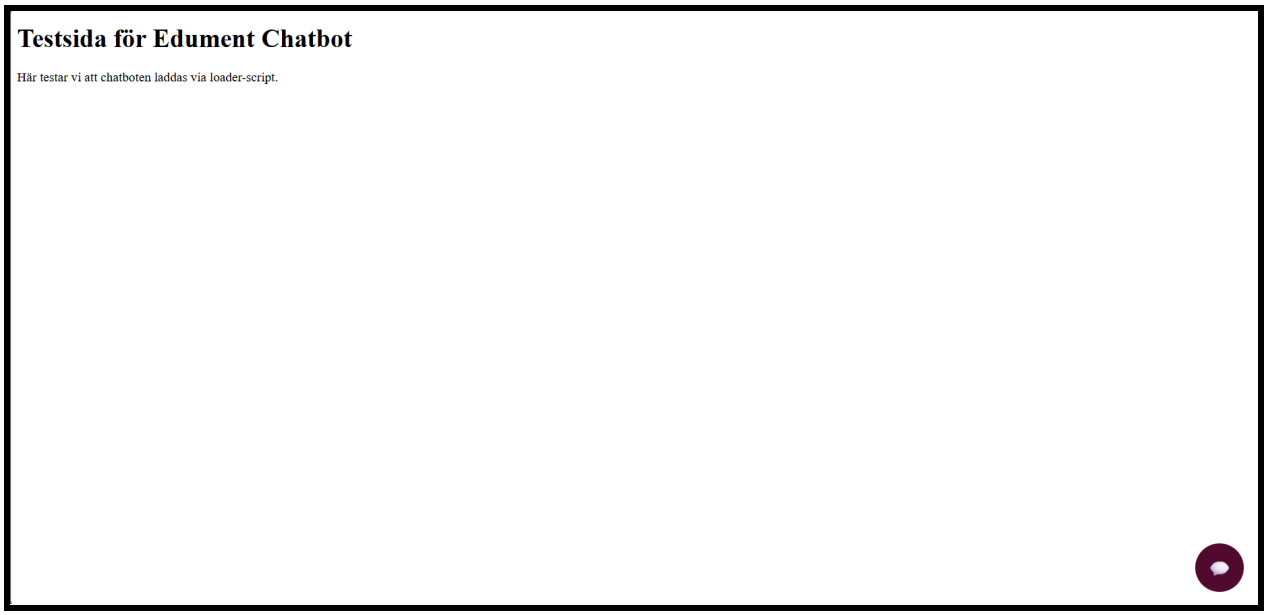


Figur 4. Bilden visar konversationen mellan användare och AI-assistenten i RAG-teknik med GPT-4o mini

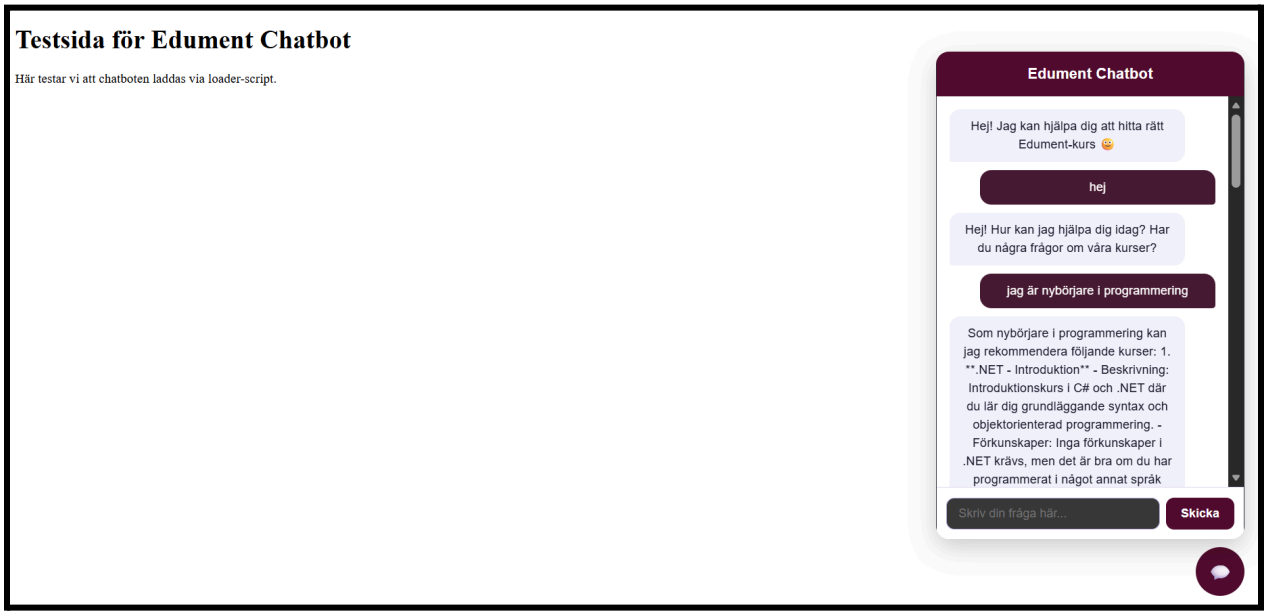
Figur 5. Bilden visar konversationen mellan användare och AI-assistenten i RAG-teknik med GPT-4o mini



Figur 6. Bilden visar konversationen mellan användare och AI-assistenten i RAG-teknik med GPT-4o mini.



Figur 7. Testsida med AI-assistents avstängd



Figur 8. Testsida med AI-assistents chatt öppen

4.4 Analys av svaren på RAG- och regelbaserade system

I examensarbetet testas två olika RAG-baserade implementationer med olika språkmodeller. Den första implementationen använder initialt phi3 via Ollama, vilket

illustreras i figur 1. Under utvecklingsarbetet testas även OpenEuroLLM-Swedish via Ollama för att förbättra den språkliga kvaliteten på svenska svar. Den slutliga implementationen använder GPT-4o mini som språkmodell, vilket visas i figur 3–6.

Resultaten visar tydliga skillnader mellan de olika tekniska lösningarna när det gäller svarens innehåll, tydlighet och anpassning till användarens frågor.

Vid frågor om specifika kurser ger det regelbaserade systemet ofta kortare och mer direkta svar. Exempelvis besvaras frågan ”När hålls ASP.Net?” i den regelbaserade implementationen med specifik information om kurstillfälle, medan implementationen med phi3 via Ollama ger generell kursinformation om kursens innehåll. Detta visar att det regelbaserade systemet i vissa fall hanterar fördefinierade frågor mer precist.

Samtidigt visar testerna att de RAG-baserade lösningarna ger större flexibilitet vid öppna frågor och varierande formuleringar av kompetensbehov. Systemet kan identifiera relevanta kursområden även när användaren inte nämner en specifik kurs direkt. Däremot observeras att svaren från phi3 inte alltid är språkligt naturliga på svenska, vilket motiverar testning av OpenEuroLLM-Swedish som alternativ språkmodell.

Resultaten visar även skillnader i svarens längd och detaljnivå. Implementationen med phi3 via Ollama genererar i vissa fall längre svar med information som inte direkt efterfrågas av användaren. Det regelbaserade systemet ger däremot kortare och mer kontrollerade svar.

Den slutliga implementationen med GPT-4o mini visar förbättringar både vad gäller språk och anpassning till användarens intention. I figur 3–6 ges exempel där AI-assistenten hanterar både kursrelaterade frågor och öppna frågor om kompetensbehov med mer naturliga och relevanta svar.

4.5 Analys av svarstid till RAG- och regelbaserade system

Tabell 2 visar uppmätta svarstider för de olika implementationerna.

Frågan/teknik	RAG-teknik med phi3 via Ollama	Regelbaseradteknik	RAG-teknik med GPT-40 mini
Jag är nybörjare i programmering	01:28 min	millisekunder	7s
Berätta gärna om c#-introduktion	01:00	01:08	10s
När hålls ASP.Net?	01:10	millisekunder	5s
Har ni react kurser?	00:56	millisekunder	12s

Resultaten visar tydliga skillnader mellan de testade lösningarna. De lokala RAG-baserade implementationerna via Ollama uppvisar svarstider på cirka en till två och en halv minut. Det regelbaserade systemet ger svar i millisekunder vid kursrelaterade frågor, medan öppna frågor som använder lokala språkmodeller ger längre svarstider.

Den slutliga implementationen med GPT-4o mini uppvisar svarstider mellan cirka fem och tolv sekunder för de testade frågorna. Samtliga testfall uppfyller därmed projektets krav på en maximal svarstid på 15 sekunder.

Resultaten visar att svarstiden varierar betydligt mellan de olika tekniska lösningarna, trots att de används för liknande typer av användarfrågor.

Svarstid är en viktig faktor för användarupplevelsen. Enligt Gnewuch et al. [9] påverkar svarstiden hur användaren upplever AI-assistenten och i vilken grad förtroende byggs upp under interaktionen. Mycket långa väntetider riskerar att skapa frustration, medan kortare svarstider bidrar till en mer effektiv användarupplevelse.

Sammantaget visar utvärderingen att den första RAG-implementationen med phi3 via Ollama hade för lång svarstid och genererade ibland svar som inte var språkligt naturliga på svenska. Det regelbaserade systemet gav snabbare och mer kontrollerade svar för väldefinierade frågor, men hade svårt att hantera öppna och varierade frågeformuleringar. Den slutliga RAG-lösningen med GPT-4o mini kombinerade god språkförståelse, flexibilitet och acceptabla svarstider, vilket motiverade valet av denna lösning.

4.6 Uppfyllda krav

Resultaten visar att examensarbetets huvudsakliga mål uppfylls genom den slutliga implementationen med GPT-4o mini. En fungerande prototyp utvecklas med både backend och frontend, där användaren kan kommunicera med AI-assistenten via ett webbgränssnitt. Systemet kan identifiera användarens kompetensbehov och ge rekommendationer baserade på Eduments kursutbud.

Vid de genomförda testerna kan systemet identifiera användarnas behov och rekommendera relevanta kurser. Samtliga externa testanvändare får korrekta rekommendationer i de testfall som genomförs. Användartesterna visar att användarna kan genomföra en fullständig interaktion med systemet via webbgränssnittet utan teknisk handledning.

När ingen lämplig kurs identifieras erbjuder systemet användaren möjlighet att fylla i ett formulär med namn och e-postadress. Användarens kontaktuppgifter och konversation vidarebefordras därefter till Edument, vilket möjliggör fortsatt dialog och hantering av behov som inte kan matchas mot det befintliga kursutbudet. Kravet på maximal svarstid uppfylls i den slutliga implementationen. Testerna visar svarstider mellan cirka fem och tolv sekunder, vilket ligger inom den uppsatta gränsen på 15 sekunder.

Sammantaget visar resultaten att den slutliga implementationen uppfyller de centrala funktionella krav som formuleras för examensarbetet och demonstrerar hur en AI-assistent kan användas för behovsanalys och vägledning inom kompetensutveckling.

5. Diskussion

5.1 Val mellan RAG- och regelbaserat system

Under examensarbetet har flera tekniska lösningar testats och jämförts för att identifiera en metod som kan kombinera god språkförståelse, relevanta kursrekommendationer och acceptabla svarstider. Utgångspunkten är projektets mål att utveckla en AI-assistent som kan identifiera användares kompetensbehov och matcha dessa mot Eduments kursutbud.

Under inlärningsfasen identifieras RAG-teknik som en lämplig metod för projektet eftersom tekniken kombinerar informationshämtning med generering av svar från en språkmodell. Detta är särskilt relevant för examensarbetet, eftersom AI-assistenten behöver basera sina rekommendationer på Eduments kursutbud snarare än på språkmodellens generella kunskap. Lewis et al. [11] beskriver att RAG använder externa kunskapskällor vid svarsgenereringen, medan AWS [5] och Microsoft Azure [4] framhåller att RAG kan minska risken för hallucinationer. På liknande sätt menar Gao et al. [12] att RAG särskilt lämpligt i domänspecifika tillämpningar där svar behöver baseras på en begränsad och kontrollerad kunskapskälla. Dessa egenskaper överensstämmer med projektets behov att matcha användare mot Eduments kursutbud.

Resultaten från den första RAG-baserade implementationen visar att tekniken i sig fungerade väl för att identifiera relevanta kursområden. Däremot framkom begränsningar i den använda språkmodellen, eftersom svaren ofta upplevdes som mindre naturliga på svenska. Detta tyder på att kvaliteten hos språkmodellen kan vara lika viktig som själva RAG-arkitekturen. Även om informationshämtningen fungerade korrekt riskerar bristande språkflyt att försämra användarupplevelsen och därmed AI-assistentens praktiska användbarhet. Enligt Wang et al. [14] utgör språkmodellen en central komponent i en AI-agent eftersom den ansvarar för tolkningen av användarens frågor.

För att förbättra den språkliga kvaliteten testas även OpenEuroLLM-Swedish via Ollama. Modellen är särskilt anpassad för europeiska språk och upplevdes generera mer naturliga formuleringar på svenska än phi3. Trots detta kvarstår problemet med långa svarstider. Detta tyder på att en förbättrad språklig kvalitet inte nödvändigtvis innebär bättre användbarhet i praktiken, eftersom användare även förväntar sig snabba svar. Det är svårt att avgöra i vilken utsträckning detta beror på själva språkmodellerna eller den hårdvara som används vid testningen. Eftersom modellerna körs lokalt utan dedikerad GPU utförs bearbetningen huvudsakligen av datorernas CPU och arbetsminne. Trots skillnader i hårdvarukonfiguration mellan testdatorerna observerades liknande svarstider vid lokal körning. Resultaten tyder därför på att den lokala körningen

av språkmodellerna hade en betydande påverkan på systemets prestanda. Samtidigt går det inte att utesluta att modellerna hade kunnat prestera bättre på kraftfullare hårdvara, vilket innebär att resultaten bör tolkas utifrån de tekniska förutsättningar som fanns i projektet.

Som ett alternativ till den RAG-baserade lösningen med lokala språkmodeller utvecklas därefter ett regelbaserat system. Resultaten visar att lösningen ger mycket korta svarstider för kursrelaterade frågor, ofta i millisekunder. Systemet är dessutom enkelt att kontrollera eftersom svaren baseras på fördefinierad logik. Detta överensstämmer med SAP [6], som beskriver att regelbaserade system ger hög precision för väldefinierade frågor och minskar risken för hallucinationer. De korta svarstiderna visar att regelbaserade system kan vara lämpliga när frågorna är välstrukturerade och förutsägbara. Samtidigt innebär den höga graden av kontroll att systemet blir mindre flexibelt än en språkmodell. Detta skapar en avvägning mellan prestanda och förmågan att förstå varierande användarbehov.

Trots dessa fördelar framträder begränsningar när systemet testas tillsammans med utvecklare på Edument. Testerna visade att det regelbaserade systemet hade svårigheter att hantera vissa formuleringar och frågetyper som inte hade identifierats under utvecklingsarbetet. Dessa begränsningar skulle delvis kunna minskas genom ytterligare användartester och utökning av regeluppsättningen. Samtidigt innebär ett sådant tillvägagångssätt ett ökat behov av löpande underhåll, eftersom nya regler behöver läggas till när nya frågetyper identifieras.

När användaren beskriver sitt kompetensbehov på ett öppet sätt räcker den regelbaserade logiken inte längre till. I dessa situationer skickas frågan vidare till en lokal språkmodell via Ollama för vidare analys. Detta innebär att även det regelbaserade systemet blir beroende av en språkmodell vid öppna frågor, vilket leder till betydligt längre svarstider än vid kursrelaterade frågor. Resultaten stödjer därmed Følstad och Brandtzægs [13] beskrivning av att regelbaserade chatbotar fungerar väl inom avgränsade domäner men har begränsad förmåga att hantera varierande användarbeteenden.

Begränsningarna hos det regelbaserade systemet, tillsammans med de långa svarstiderna hos de lokalt körda språkmodellerna, ledde till att GPT-4o mini utvärderades efter rekommendation från Edument. Resultaten visar att modellen kombinerar god språkförståelse med betydligt kortare svarstider än de lokala språkmodellerna. Enligt Gnewuch et al. [9] är svarstid en viktig faktor för användarupplevelse och förtroende för AI-baserade system. Detta gör de observerade skillnaderna särskilt betydelsefulla, eftersom långa väntetider riskerar att försämra användarens upplevelse även om svaren är korrekta. GPT-4o mini uppfyller samtidigt

kravet på maximal svarstid och behåller den flexibilitet som RAG-tekniken erbjuder vid hantering av öppna kompetensbehov. Till skillnad från det regelbaserade systemet kan modellen hantera varierande formuleringar och mer komplexa användarbehov, samtidigt som svarstiden är betydligt kortare än för de lokalt körda språkmodellerna.

Sammantaget visar resultaten att de undersökta lösningarna har olika styrkor och svagheter. Å ena sidan erbjuder det regelbaserade systemet hög precision och mycket korta svarstider för väldefinierade frågor. Å andra sidan begränsas dess användbarhet av att systemet har svårt att hantera varierande formuleringar och öppna kompetensbehov. RAG-baserade lösningar erbjuder däremot större flexibilitet eftersom de kan tolka användarens behov även när frågorna inte följer förutbestämda mönster. Samtidigt visade resultaten att prestandan hos de lokalt körda språkmodellerna påverkades negativt av långa svarstider. Mot denna bakgrund framstår GPT-4o mini som den lösning som bäst balanserar språkförståelse, svarskvalitet, flexibilitet och svarstid. Därför bedöms denna lösning vara mest lämplig för projektets syfte och de krav som ställts på AI-assistenten.

5.2 Behovsanalys och kursmatchning

En central del av examensarbetet är AI-assistentens förmåga att identifiera användarens kompetensbehov och matcha dessa mot relevanta kurser. Resultaten visar att kvaliteten på både behovsanalysen och kursmatchningen varierar mellan de testade lösningarna.

Den första RAG-baserade implementationen med phi3 visar att systemet kan identifiera relevanta kursområden, men att behovsanalysen inte alltid blir tillräckligt träffsäker. I figur 1 besvaras exempelvis frågan ”När hålls ASP.Net?” med generell kursinformation istället för information om kurstillfälle. Resultatet visar att systemet hittar relevant kursrelaterad information men inte fullt ut tolkar användarens egentliga intention. Detta påverkar även kursmatchningen eftersom rekommendationen bygger på hur väl användarens behov förstås.

Liknande begränsningar observeras i det regelbaserade systemet. Externa tester tillsammans med utvecklare på Edument visar att systemet fungerar väl för fördefinierade kursfrågor men har svårare att hantera varierande formuleringar och underförstådda behov. I flera testfall har systemet svårt att tolka användarens underliggande intention, vilket leder till ofullständiga svar eller mindre relevanta rekommendationer. Detta illustreras i figur 2, där systemet begränsar möjligheten att genomföra en djupare behovsanalys när användaren inte uttrycker sitt behov på ett sätt som överensstämmer med den fördefinierade logiken.

Den slutliga implementationen med GPT-4o mini visar däremot en tydligt förbättrad förmåga att förstå användarens intention och hantera öppna frågor om kompetensutveckling. Jämförelsen mellan figur 1 och figur 3 visar att GPT-4o mini i högre grad kan tolka sammanhanget i användarens fråga och generera svar som är bättre anpassade till det efterfrågade behovet. Detta bidrar till en mer träffsäker behovsanalys och därmed även bättre kursmatchning.

Kursrekommendationernas relevans verifieras genom att de genererade svaren jämförs med kursinformationen i både JSON-filerna och Eduments webbplats. Resultaten från de genomförda testerna visar att GPT-4o mini rekommenderar kurser som överensstämmer med den information som hämtas från kursunderlaget via RAG-komponenten. Eftersom kursinformationen hämtas från den externa kunskapsbasen kan kursunderlaget uppdateras utan att språkmodellen behöver tränas om. Externa tester tillsammans med utvecklare på Edument bekräftar även att rekommendationerna upplevs som mer relevanta och flexibla än de som genereras av det regelbaserade systemet.

Resultaten visar också att behovsanalysen inte enbart handlar om att rekommendera en kurs. I situationer där ingen lämplig kurs identifieras erbjuder systemet användaren ett kontaktformulär, vilket illustreras i figur 8. Genom formuläret kan användaren ange namn och e-postadress, varefter kontaktuppgifterna och konversationen mellan användaren och AI-assistenten vidarebefordras till Edument. Funktionen möjliggör att användarens kontaktuppgifter och behov kan vidarebefordras till Edument även i de fall där systemet inte identifierar en lämplig kursrekommendation.

Sammantaget visar resultaten att GPT-4o mini ger den mest träffsäkra behovsanalysen och kursmatchningen av de testade lösningarna. Kombinationen av förbättrad språkförståelse, flexibilitet och möjligheten att hantera situationer där ingen kurs matchar användarens behov gör att den slutliga lösningen bättre stödjer både användaren och Eduments verksamhet.

5.3 Betydelsen av svarstid och användarförtroende

Svarstid visar sig vara en av de viktigaste faktorerna för användarupplevelsen i examensarbetet. Även om en AI-assistent kan ge korrekta och relevanta svar påverkas användarens upplevelse negativt om väntetiden blir för lång. Resultaten visar att de lokala språkmodellerna phi3 och OpenEuroLLM-Swedish ofta genererar svarstider på mellan en och två och en halv minut, medan vissa öppna frågor i det regelbaserade systemet kan ta upp till åtta minuter eftersom de skickas vidare till en lokal språkmodell

för analys. Dessa väntetider bedöms vara långa i relation till moderna digitala tjänster och riskerar att få användaren att tro att systemet har slutat fungera eller att ett tekniskt fel har uppstått.

Den slutliga implementationen med GPT-4o mini uppvisar svarstider på cirka fem till tolv sekunder. Resultaten visar att dessa svarstider upplevs som betydligt mer acceptabla än väntetider på flera minuter. Samtidigt observeras att även väntetider över tio sekunder börjar påverka användarupplevelsen negativt. Detta indikerar att korta svarstider är viktiga för användarupplevelsen, särskilt i jämförelse med moderna digitala tjänster där information ofta presenteras nästan omedelbart.

Wang och Lo (2025) beskriver att svarstid fungerar som en social signal som påverkar hur användare uppfattar en AI-assistent. En mycket lång väntetid kan minska användarens förtroende för systemet, medan en rimlig svarstid kan bidra till en mer positiv användarupplevelse. Resultaten från examensarbetet stödjer denna observation, eftersom användarnas upplevelse förbättras tydligt när svarstiden minskar från minuter till sekunder.

Samtidigt visar resultaten att svarstid inte är den enda faktorn som påverkar användarupplevelsen. Under projektet bedöms relevansen och kvaliteten på svaren vara viktigare än att svaren genereras så snabbt som möjligt. Ett snabbt men irrelevant svar bidrar inte till att lösa användarens problem. GPT-4o mini visar därför en fördel genom att kombinera relativt korta svarstider med god språkförståelse och relevanta rekommendationer.

Detta resonemang överensstämmer delvis med Gnewuch et al. (2018), som argumenterar för att snabbare svar inte alltid leder till en bättre upplevelse. Författarna visar att användare i vissa situationer kan uppfatta måttliga svarsfördröjningar som mer naturliga än omedelbara svar. Resultaten från examensarbetet tyder dock på att det finns en tydlig gräns där väntetiden blir så lång att den påverkar användarens förtroende och vilja att fortsätta interaktionen.

Svarstid påverkar även användarförtroendet indirekt genom kvaliteten på de svar som genereras. AWS [5], Microsoft Azure [4] och Gao et al. [12] framhåller att RAG-teknik kan öka trovärdigheten genom att basera svar på externa kunskapskällor istället för enbart språkmodellens interna kunskap. När användaren får relevanta och välgrundade svar ökar sannolikheten att systemet uppfattas som tillförlitligt. Resultaten visar därför att användarförtroende påverkas av en kombination av svarstid, svarskvalitet och systemets förmåga att förstå användarens behov.

Som framgår av figur 3–6 genererar GPT-4o mini relevanta och språkligt naturliga svar samtidigt som svarstiden uppfyller det uppsatta kravet på maximalt 15 sekunder.

Jämfört med de lokalt körda språkmodellerna indikerar resultaten att en balans mellan svarskvalitet och svarstid är viktig för att skapa en positiv användarupplevelse. GPT-4o mini bedöms ge den bästa balansen mellan dessa faktorer och har därför störst potential att bidra till ett ökat användarförtroende bland de testade lösningarna.

5.4 Frontend och användargränssnitt

Frontendens utformning spelar en viktig roll för hur användaren upplever AI-assistenten. Även om den bakomliggande tekniken kan generera korrekta svar och relevanta rekommendationer är det avgörande att användaren enkelt kan förstå hur systemet ska användas. En välfungerande AI-assistent handlar därför inte enbart om språkmodeller och algoritmer, utan även om hur information presenteras för användaren.

Resultaten visar att det utvecklade användargränssnittet upplevs som enkelt, tydligt och lätt att använda. Under testerna kan användarna genomföra hela processen från att beskriva sitt kompetensbehov till att ta del av kursrekommendationer utan teknisk handledning. Detta indikerar att gränssnittet stödjer användarens interaktion med systemet på ett effektivt sätt.

Figur 3–8 illustrerar de olika stegen i användarflödet. Användaren kan ställa frågor, få rekommenderade kurser, ta del av kursinformation och i förekommande fall lämna kontaktuppgifter för fortsatt dialog med Edument. Särskilt kontaktformuläret bidrar till användarupplevelsen genom att erbjuda ett alternativ när ingen lämplig kurs identifieras. Istället för att interaktionen avslutas kan användaren fortsätta dialogen genom att skicka sina kontaktuppgifter tillsammans med konversationen till Edument.

Användargränssnittet bidrar även till att göra AI-assistentens funktioner mer tillgängliga. Behovsanalys, kursmatchning och kontaktförmedling integreras i samma gränssnitt, vilket skapar ett sammanhängande användarflöde. Detta minskar risken för att användaren behöver växla mellan olika system eller kommunikationskanaler för att få hjälp.

Samtidigt identifieras möjliga förbättringsområden. Ett exempel är stöd för röstinmatning eller inspelningsfunktioner, vilket skulle kunna göra systemet mer tillgängligt för användare som föredrar att beskriva sina behov muntligt istället för skriftligt. Detta skulle kunna bidra till en mer naturlig interaktion och utgör därför ett intressant område för vidare utveckling.

Sammantaget visar resultaten att frontendens enkelhet och tydlighet bidrar till en positiv användarupplevelse. Kombinationen av ett lättanvänt gränssnitt, relevanta kursrekommendationer och möjligheten till fortsatt kontakt med Edument stärker systemets användbarhet och stödjer examensarbetets övergripande mål.

5.5 Databehandling och transparens

En annan aspekt gäller hur användardata behandlas. Den slutliga lösningen skickar användarens frågor till en extern tjänst via OpenAI API, medan de lokala modellerna bearbetade informationen lokalt. Detta kan innebära andra krav på transparens och information till användaren kring hur data behandlas. Inom ramen för examensarbetet har användarnas inställning till denna fråga inte undersökts närmare, men det kan vara relevant att beakta i framtida utveckling av systemet.

6. Slutsats

6.1 Sammanfattning

Syftet med examensarbetet är att utveckla en AI-assistent som kan stödja behovsanalys och vägledning inom kompetensutveckling genom att identifiera användares behov och matcha dessa mot relevanta kurser. Resultaten visar att detta syfte uppfylls genom den slutliga implementationen baserad på RAG-teknik och GPT-4o mini.

Under projektets gång utvärderas flera tekniska lösningar, inklusive RAG med lokala språkmodeller, ett regelbaserat system samt en slutlig RAG-baserad lösning med GPT-4o mini. Resultaten visar att kombinationen av RAG och GPT-4o mini ger den bästa balansen mellan språkförståelse, behovsanalys, kursmatchning och svarstid. Lösningen kan hantera både kursrelaterade frågor och öppna kompetensbehov samt rekommendera relevanta kurser baserat på Eduments kursutbud.

Arbetet visar även att valet av språkmodell har stor betydelse för både användarupplevelse och kvaliteten på behovsanalysen. De lokala språkmodellerna phi3 och OpenEuroLLM-Swedish uppvisar begränsningar i form av långa svarstider, medan phi3 dessutom genererar svar som inte alltid är språkligt naturliga på svenska. Resultaten indikerar även att hårdvarans prestanda, särskilt CPU och arbetsminne, kan påverka svarstiderna vid lokal körning av språkmodeller.

Det regelbaserade systemet visar sig vara snabbt och precist för väldefinierade frågor, men har begränsad förmåga att hantera öppna frågor och varierande formuleringar av användarbehov. Detta gör att systemet blir mindre lämpligt som ensam lösning för behovsanalys inom kompetensutveckling.

Den slutliga lösningen bidrar till Eduments verksamhet genom att erbjuda automatiserad behovsanalys, kursmatchning och hantering av situationer där ingen

lämplig kurs kan identifieras. Genom kontaktformuläret kan användarens kontaktuppgifter och konversation vidarebefordras till Edument för fortsatt dialog, vilket säkerställer att användarens behov kan hanteras även när en direkt kursmatchning saknas.

6.2 Reflektion över etiska aspekter

Flera etiska aspekter aktualiseras under utvecklingen av den AI-assistent som tas fram i examensarbetet. De mest relevanta aspekterna handlar om personuppgifter, datainsamling och hantering av information som delas av användaren.

I den slutgiltiga lösningen erbjuds användaren möjlighet att fylla i ett kontaktformulär när ingen lämplig kurs kan identifieras. Genom formuläret samlas namn och e-postadress in och skickas tillsammans med konversationen till Edument för fortsatt hantering. Detta innebär att personuppgifter behandlas av systemet, vilket ställer krav på att användaren informeras om hur uppgifterna används och vem som får tillgång till dem.

En annan aspekt rör datainsamling och datalagring. Användaren kan i sina frågor dela information om arbetsroll, kompetensbehov eller andra uppgifter som kan vara personliga eller verksamhetsrelaterade. Även om syftet med insamlingen är att möjliggöra bättre behovsanalys och kursrekommendationer bör endast den information som är nödvändig samlas in och hanteras.

Sekretess är också en viktig aspekt. Eftersom användaren fritt kan formulera sina frågor finns det en risk att konfidentiell information eller känsliga uppgifter delas i konversationen. Det är därför viktigt att användare är medvetna om vilken typ av information som bör undvikas vid användning av systemet.

Samtidigt kan AI-assistenten bidra med samhällsnytta genom att förenkla tillgången till kompetensutveckling och göra det enklare för individer och organisationer att identifiera relevanta utbildningar. Genom att effektivisera den inledande behovsanalysen kan systemet bidra till snabbare och mer tillgänglig vägledning inom kompetensutveckling.

Sammantaget visar examensarbetet att utvecklingen av AI-baserade system inte enbart handlar om tekniska lösningar. Frågor som rör personuppgifter, datainsamling, sekretess och ansvarsfull användning av information behöver också beaktas för att skapa ett system som används på ett etiskt och förtroendefullt sätt.

6.3 Framtida arbete

Detta examensarbete har gett en grundläggande förståelse för hur AI-assistenter kan utvecklas och användas inom kompetensmatchning och utbildning. För framtida studenter på LTH finns flera möjliga områden att vidareutveckla och undersöka.

Ett möjligt framtida arbete är att förbättra precisionen i RAG-system genom att jämföra olika embeddings-modeller och vektordatabaser. Det skulle kunna ge en djupare förståelse för hur semantisk sökning påverkar kvaliteten på AI-genererade svar.

En annan möjlig fortsättning är att undersöka skillnader mellan lokala språkmodeller och molnbaserade språkmodeller ur perspektiv som prestanda, kostnad, säkerhet och energiförbrukning. I detta examensarbete gav GPT-4o mini bättre resultat än phi3 via Ollama, men lokala modeller kan vara intressanta ur integritets- och hållbarhetssynpunkt.

Framtida studenter skulle även kunna utveckla mer avancerade AI-agenter med minnesfunktioner och förbättrad kontextförståelse. Detta skulle göra det möjligt för AI-assistenten att hålla längre och mer naturliga konversationer med användaren.

Ett annat intressant område är att undersöka hur AI-assistenter påverkar användarförtroende och användarupplevelse. Exempelvis kan framtida arbeten analysera hur svarstid, språkstil och graden av “mänsklig” kommunikation påverkar hur användare upplever systemet.

Det finns även möjlighet att vidareutveckla frontend-lösningen genom att skapa mer avancerade och interaktiva användargränssnitt. Framtida projekt skulle kunna undersöka integration med mobilapplikationer eller röststyrda AI-assistenter.

Slutligen kan framtida examensarbeten fokusera mer på etiska och juridiska aspekter kring AI-system, exempelvis GDPR, transparens och ansvarsfull användning av generativ AI i verkliga verksamheter.

7. Källor

[1] Tech with Tim (2024), *How to Build a Local AI Agent With Python (Ollama, LangChain & RAG)*, YouTube,
YouTube (Hämtad 2026-03-10).

[2] Pandas Development Team (u.å.), *Pandas Documentation*,
Pandas Documentation
(Hämtad 2026-04-19).

[3] Edument AB (u.å.), *Utbildningar*,
Edument Training
(Hämtad 2026-03-3).

[4] Microsoft Azure (u.å.), *What is Retrieval-Augmented Generation (RAG)?*,
Microsoft Azure – What is Retrieval-Augmented Generation (RAG)?
(Hämtad 2026-04-28).

[5] Amazon Web Services (u.å.), *What is Retrieval-Augmented Generation?*,
AWS – What is Retrieval-Augmented Generation?
(Hämtad 2026-04-28).

[6] SAP (u.å.), *What is a Chatbot?*,
SAP – What is a chatbot?
(Hämtad 2026-04-28).

[7] Vite (u.å.), *Getting Started*,
Vite Documentation
(Hämtad 2026-05-5).

[8] Pallets Projects (u.å.), *Flask Documentation*,
Flask Documentation
(Hämtad 2026-05-5).

[9] Gnewuch, U., Morana, S., Adam, M. T. P. och Maedche, A. (2018), *Faster Is Not Always Better: Understanding the Effect of Dynamic Response Delays in Human-Chatbot Interaction*, Proceedings of the 26th European Conference on Information Systems (ECIS 2018).

[10] European Commission (2019), *Ethics Guidelines for Trustworthy AI*,
European Commission – Ethics Guidelines for Trustworthy AI
(Hämtad 2026-05-8).

[11] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S. och Kiela, D. (2020), *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*, NeurIPS 2020.

[12] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Wang, Y., Sun, J., Wang, M. och Wang, H. (2023), *Retrieval-Augmented Generation for Large Language Models: A*

Survey, arXiv,
https://doi.org/10.48550/arXiv.2312.10997

[13] Følstad, A. och Brandtzæg, P. B. (2017), *Chatbots and the New World of HCI*, Interactions, 24(4), s. 38–42.

[14] Wang, L., Ma, C., Feng, X. et al. (2024), *A Survey on Large Language Model Based Autonomous Agents*, Frontiers of Computer Science, 18(6), 186345.
https://doi.org/10.1007/s11704-024-40231-1

[15] Huang et al. (2025). *The effects of response time on older and young adults' interaction experience with Chatbot*.

The effects of response time on older and young adults' interaction experience with Chatbot - PMC

[16] GPT-4o mini: Vi utvecklar kostnadseffektiv intelligens GPT-4o mini: Vi utvecklar kostnadseffektiv intelligens | OpenAI

(Hämtad 2026-05-26)

[17]ChatOpenAI integration - Docs by LangChain (hämtad 2026-05-26)

[18] https://www.ollama.com/jobautomation/OpenEuroLLM-Swedish (hämtad 2026-05-29)

[19] Ranjan, A. (2025). How Chatbots Influence Customer Experience and Conversion Rates in E-Commerce. *IRE Journals*, 8(11), 2267–2268.

https://www.irejournals.com/paper-details/1708825

(Hämtad 2026-06-01)

[20] Bergmann, D. och Stryker, C. (u.å.). *What is LangChain?*. IBM.

https://www.ibm.com/think/topics/langchain?utm_source=chatgpt.com

(Hämtad 2026-06-01)